АиСД у2024. Первый семестр

Домашние задания М3134-М3135

⟨Версия от 24 сентября 2024 г.⟩

Темы

Неделя 1. Введение. Асимптотические оценки алгоритмов.

1.1. Докажите по определению, что если $f_1(n) = \mathcal{O}(g_1(n))$ и $f_2(n) = \mathcal{O}(g_2(n))$, то $f_1(n) + f_2(n) = \mathcal{O}(g_1(n) + g_2(n))$, если $f_i(n)$ и $g_i(n)$ неотрицательны.

1.2. Докажите по определению, что $\max(f(n), g(n)) = \Theta(f(n) + g(n))$, если $f(n)$ и $g(n)$ неотрицательны.

1.3. Докажите по определению, что $\sum\limits_{i=1}^{n+5} 2^i = \mathcal{O}(2^n)$.

1.4. Докажите по определению, что $\frac{n^3}{6} - 7n^2 = \Omega(n^3)$.

1.5. Докажите по определению, что $2^n \cdot n^2 = \mathcal{O}(2.1^n)$.

1.6. Докажите по определению, что $\log_3(n^2 + 5n + 3) = \Theta(\log n)$.

1.7. Докажите или опровергните следующие утверждения. Для опровержения достаточно привести пример любой функции (не обязательно должна быть сложность какого-либо алгоритма), на котором утверждение неверно. Все функции положительные.

(a) Если $f(n) = \mathcal{O}(g(n))$, то $g(n) = \mathcal{O}(f(n))$

(b) Если $f(n) = \mathcal{O}(g(n))$, то $2^{f(n)} = \mathcal{O}(2^{g(n)})$

(c) $f(n) = \mathcal{O}(f(n)^2)$

(d) $f(n) = \Theta(f(\frac{n}{2}))$

(e) Если $f(n) = \mathcal{O}(g(n))$, то $g(n) = \Omega(f(n))$

1.8. Пусть $p(n) = \sum\limits_{i=0}^{d} a_i n^i$, где $a_d > 0$ — полином степени d от n. Используя определения, полагая, что k — константа, докажите, что:

(a) Если $k \geq d$, то $p(n) = \mathcal{O}(n^k)$

(b) Если $k = d$, то $p(n) = \Theta(n^k)$

(c) Если $k \leq d$, то $p(n) = \Omega(n^k)$

1.9. Придумайте две положительные функции $f(n)$ и $g(n)$, такие что не выполнено ни одно из двух: $f(n) = \mathcal{O}(g(n))$ и $f(n) = \Omega(g(n))$.

1.10. Докажем, что $n^2 = \mathcal{O}(n)$ по индукции. Для этого докажем, что для любого k верно, что $kn = \mathcal{O}(n)$. Действительно, для $k = 1$ это верно. Пусть утверждение верно для $k-1$, тогда $kn = (k-1)n + n = \mathcal{O}(n)$. Таким образом, это верно для всех k, а значит и для $k = n$, таким образом $n \cdot n = \mathcal{O}(n)$. Где ошибка в этом доказательстве?

1.11. Докажите или опровергните, что $\log(n!) = \Theta(n \log n)$.

1.12. Докажите или опровергните, что $\sum\limits_{i=1}^{n} \frac{n}{i} = \mathcal{O}(n \log n)$.

1.13. Докажите по индукции, что если $T(n) = 2T(\sqrt{n}) + 1$, то $T(n) = \mathcal{O}(\log n)$. В этом и следующих заданиях в качестве базы индукции можно полагать, что $T(1) = 1$.

1.14. Докажите по индукции, что если $T(n) = 3T\left(\frac{n}{2}\right) + 1$, то $T(n) = \Omega(n)$.

1.15. Докажите по индукции, что если $T(n) = \log n \cdot T\left(\frac{n}{\log n}\right) + n$, то $T(n) = \mathcal{O}(n \log n)$.

1.16. Докажите по индукции, что если $T(n) = 2T\left(\frac{n}{2} + \log n\right) + n$, то $T(n) = \mathcal{O}(n \log n)$.

1.17. Докажите по индукции, что если $T(n) = 2T\left(\frac{n}{2} + 20\right) + n$, то $T(n) = \mathcal{O}(n \log n)$.

1.18. Найдите Θ-асимптотику времени работы алгоритма, построив дерево рекурсии, если $T(n) = T\left(\frac{n}{3}\right) + \log_2(n)$.

1.19. Доказательство Мастер-теоремы. Пусть $T(n) = aT\left(\frac{n}{b}\right) + n^c$. Докажите, что:

(a) Если $c < \log_b a$, то $T(n) = \mathcal{O}(n^{\log_b a})$

(b) Если $c > \log_b a$, то $T(n) = \mathcal{O}(n^c)$

(c) Если $c = \log_b a$, то $T(n) = \mathcal{O}(n^c \log n)$

Подсказка. Для начала оцените количество запусков на каждом уровне рекурсии, а также размер задачи (то есть, n) для каждого такого запуска. После этого оцените суммарное количество совершенных действий на каждом уровне рекурсии и просуммируйте. Полученная сумма в каждом из трех случаев должна красиво свернуться.

1.20. Пусть время работы алгоритма A равно $T_A(n) = 7T_A\left(\frac{n}{2}\right) + n^2$, а время работы алгоритма B равно $T_B(n) = aT_B\left(\frac{n}{4}\right) + n$. При каких значениях a второй алгоритм работает асимптотически быстрее первого?

Время работы некоторого алгоритма задано следующим рекуррентным соотношением. Найдите Θ-асимптотику времени работы этого алгоритма, *построив дерево рекурсивных вызовов.*

1.21. $T(n) = T(n-1) + n$

1.22. $T(n) = 3T\left(\frac{n}{2}\right) + 3$

1.23. $T(n) = T\left(\frac{n}{3}\right) + \log n$

Неделя 2. Квадратичные сортировки. Сортировка слиянием. Два указателя

2.24. Даны k отсортированных массивов, сумма длин которых равна n. Придумайте алгоритм, позволяющий объединить все эти массивы в один отсортированный массив. Время работы $\mathcal{O}(nk)$.

2.25. Даны два отсортированных массива a и b длины n. Найдите такие i и j, что значение $|a_i - b_j|$ минимально. Время работы $\mathcal{O}(n)$.

2.26. Даны два отсортированных массива a и b длины n, а также число S. Найдите такие i и j, что $a_i + b_j = S$. Время работы $\mathcal{O}(n)$.

2.27. Даны два отсортированных массива a и b длины n. Найдите количество пар (i, j), таких что $a_i = b_j$. Время работы $\mathcal{O}(n)$.

2.28. Дан массив a длины n, состоящий из неотрицательных целых чисел. Найдите в данном массиве подотрезок $[l, r]$, такой что $a_l + a_{l+1} + \ldots + a_r = k$ либо определите, что его не существует. Время работы $\mathcal{O}(n)$.

2.29. Решите предыдущую задачу при условии, что нужно найти подотрезок максимальной длины.

2.30. Дан массив a длины n, состоящий из неотрицательных целых чисел. Найдите количество подотрезков массива, сумма которых равна k.

2.31. Дан массив a. Назовем *инверсией* пару индексов (i, j), такую что $i < j$ и $a_i > a_j$. Пусть в массиве длины n ровно k инверсий. Докажите, что сортировка вставками работает за $\mathcal{O}(n + k)$.

2.32. Дан массив a длины n. Найдите количество инверсий в массиве за $\mathcal{O}(n \log n)$.

Подсказка. Модифицируйте сортировку слиянием.

2.33. Придумайте способ сделать сортировку слиянием «снизу вверх», без использования рекурсии.

2.34. В отсортированный массив длины n в произвольное место вставили новый элемент. Придумайте алгоритм, позволяющий отсортировать получившийся массив за $\mathcal{O}(n)$.

2.35. Постройте для произвольного n перестановку, для которой сортировка вставками сделает наибольшее количество операций `swap`. Сколько в точности обменов совершится? Единственна ли построенная перестановка?

2.36. Дан массив a длины n. Необходимо для каждого элемента найти количество элементов, меньших его. Время работы $\mathcal{O}(n \log n)$.

2.37. Дан массив a длины n, состоящий из натуральных чисел. Требуется найти минимальное натуральное число, которого нет в массиве, за $\mathcal{O}(n \log n)$.

2.38. Даны два массива a и b длины n. Найдите такую пару (i, j), что $a_i < a_j$ и $b_i < b_j$, либо определите, что такой пары не существует. Время работы $\mathcal{O}(n \log n)$.

2.39. Сортировка называется *стабильной*, если она сохраняет порядок элементов, не различимых при помощи операции сравнения. Например, если мы хотим отсортировать массив пар чисел, а при сортировке мы сравниваем пары только по первым элементам, если сортировка является стабильной, массив $[(1,3),(0,4),(1,0),(0,5)]$ будет отсортирован следующим образом: $[(0,4),(0,5),(1,3),(1,0)]$. Проанализируйте сортировки пузырьком, выбором, вставками и слиянием, и выясните, являются ли они стабильными.

2.40. Докажите, что любую сортировку можно сделать стабильной, потратив $\mathcal{O}(n)$ дополнительной памяти, не меняя время работы сортировки.

2.41. Дан массив a длины n. Постройте массив b длины n, состоящий из чисел от 0 до $n-1$, чтобы было выполнено следующее свойство: если $a_i < a_j$, то $b_i < b_j$. Время работы $\mathcal{O}(n \log n)$.